ECE 517: Machine Learning

Assignment 6.1: Primal Formulation of a Nonlinear Classifier With MMSE

David Kirby – 101652098 – davidkirby@unm.edu

Fall 2021

PRIMAL FORMULATION OF A NONLINEAR CLASSIFIER WITH MMSE

Use the attached function to reconstruct the example of lesson 6.1. In particular, you must:

1. Construct a train dataset and represent them. Your representation may be rotated 90° with respect to the one of the slides.
2. Construct a function to directly map the data into a 10 dimension Hilbert space using the Volterra expansion.
3. Compute the weights of the MMSE solution, and represent the boundary as indicated in the slides.

Provide a document that summarizes the theory and a graph of the result. Comment your results.

[Code omitted for brevity.]

Primal formulation of a non-linear classifier with MMSE.

This problem takes a linear channel and solves it using a non-linear approach consisting of transforming our data into a higher dimensional Hilbert space (a higher dimensional vector space). One way to do this is with the Volterra expansion which computes products of the different features of our vector. We have a vector of two components, which in our case is $x[n]$ and $x[n-1]$, and we compute combinations of products up to a third order. With this, we have ten dimensions, including one where we have a cost. Then, we put all these components into a new vector of ten components and we apply the Minimum Mean Square Error approach. So, $\mathbf{w}$ has the same solution as in our previous lectures, but instead of having $\mathbf{x}_n$, we have Φ, where Φ is a matrix containing all the values of the transformed vector.

$$\mathbf{w} = (\Phi\Phi^\top)^{-1}\Phi y \tag{1}$$

Adjusting the parameters following this algorithm, we can represent points defined by the hyperplane $\mathbf{w}^\top\phi(x_n) = 0$ and these points are the boundary (as shown in Figure 1) that non-linearly classify almost all points.

The problem is that we have ten dimensions for an input of two dimensions. For example, if we use an expansion with order five, then we would need 56 elements, and so on. This is what we call the curse of dimensionality. If we want a solution that works better than this example, which is not very efficient, we need to add more expressive capacity to our machine by using a space with more dimensions. In other words, we have to increase the Vapnik–Chervonenkis dimension on that space.

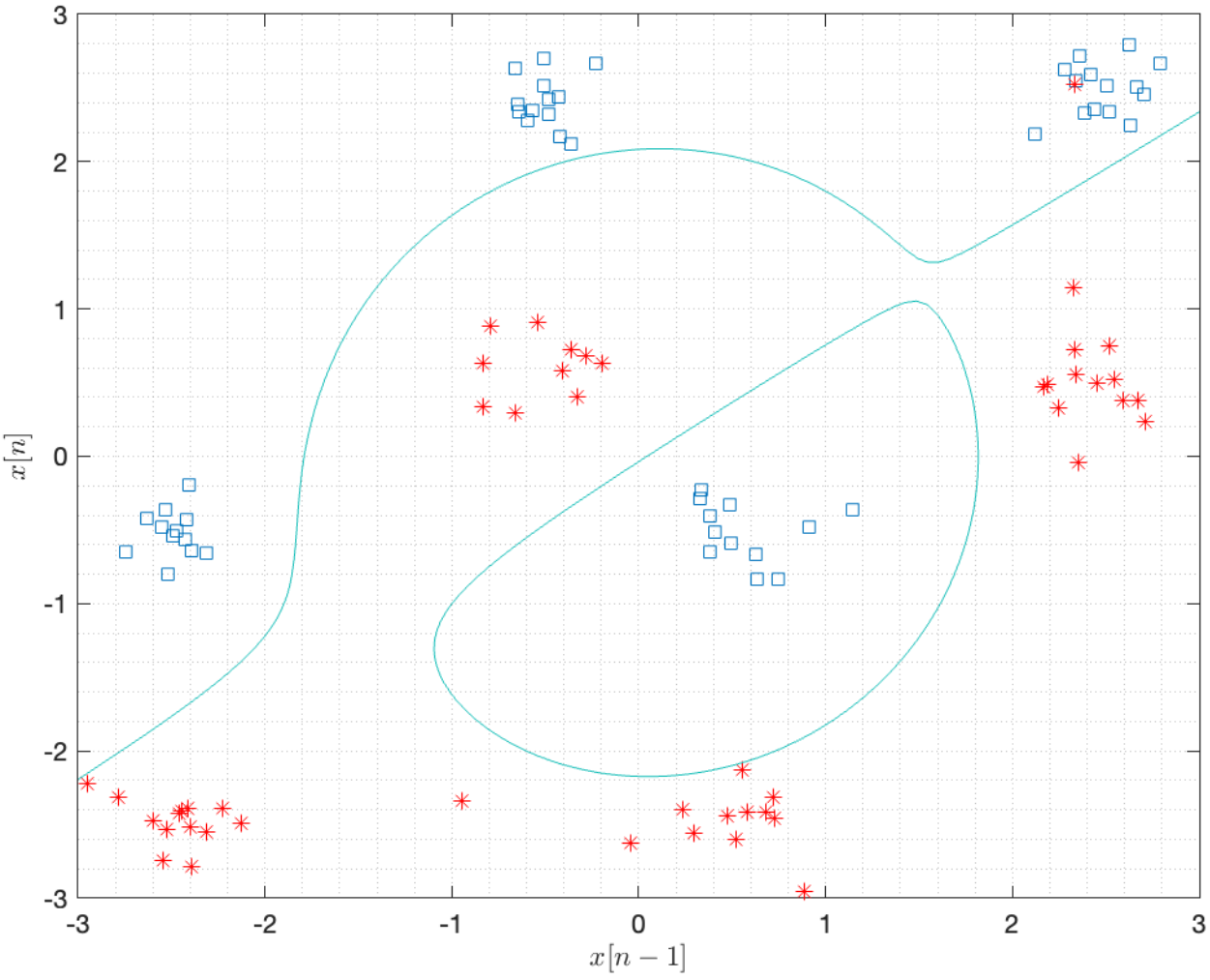

Figure 1: Computed weights of the MMSE solution and representation of the boundary.